ESZTERHÁZY KÁROLY KATOLIKUS EGYETEM

Matematikai és Informatikai Intézet

A szakdolgozat címe

Készítette
Bartus János
Programtervező informatikus

Témavezető
Troll Ede
Tanársegéd

Eger, 2025

Tartalomjegyzék

Bevezetés

Gyerekkorom óta érdekeltek a videójátékok, már egészen kicsiként kezdtem el játszani. Az első komolyabb élményeimet apukám PlayStation 2 konzolján szereztem ahol sok időt töltöttem különböző játékokkal. Akkoriban még csak a játék öröme vonzott, nem is gondoltam volna hogy ennyire mélyen elmerülök ebben a világban. Ahogy idősebb lettem, általános iskola alsó tagozatában kezdtek jobban érdekelni a videójátékok nemcsak mint szórakozás hanem mint rendszerek. Elkezdtem azon gondolkodni hogyan működnek, mi és hogy irányítja a karaktereket és a játékon belüli eseményeket.

A programozás irányába a Minecraft vezetett. Felfedeztem benne a parancsblokkokkat és azokkal próbáltam változtatni a játék világát. Egyre izgalmasabbá vált hogy a saját ötleteimet meg tudom valósítani, ekkor határoztam el hogy programozó akarok lenni.

A programozással középiskolában kezdtem el komolyabban foglalkozni. Ekkor már tudatosan kerestem azokat a lehetőségeket, amelyekkel jobban megérthetem hogyan működnek különböző játékok és szoftverek. Az első lépéseimet kisebb programok megírásával kezdtem. Kezdetben iskolai beadandók során próbáltam ki magam, de rájöttem hogy a játékfejlesztés felé saját projektekkel lehet jobban elmélyedni. Ezért elkezdtem kisebb játékokat és interaktív szoftvereket fejleszteni, amivel nemcsak a programozás alapjait tudtam gyakorolni, hanem a játékok logikáját is elkezdtem jobban megérteni.

Komolyabban foglalkozni a programozással egyetem alatt kezdtem el. Ebben az időszakban rengeteg új dolgot tanultam, és megismerkedtem számos különböző programozási nyelvvel és technikával. Az egyetem nemcsak az elméleti tudást adta meg, hanem lehetőséget adott, hogy gyakorlati tapasztalatot szerezzek különböző feladatok és projektek révén.

A számomra eddig ismeretlen platformer játék stílussal is az egyetem alatt ismerkedtem meg. Az első találkozásom ezzel a műfajjal izgalmas kihívások elé állított amely lehetőséget adott arra, hogy új területeken is kipróbáljam magam a játékfejlesztésben. A platformerek különlegessége számomra az volt, hogy egyesíteni kell bennük a szórakoztató játékmechanikát, a kreatív pályadizájnt és az erőteljes történetmesélést. Az igazi áttörés az egyetem megrendezésre kerülő GémDzsem alatt tapasztam. Olyan hatással volt rám ez az élmény hogy eldöntöttem: szeretnék a játékfejlesztéssel komolyan foglalkozni.

A szakdolgozatom során először készítek platformer játékot, amelynek animációit és grafikáit egy grafikus tervezi és készíti el. Ez új kihívást jelent számomra, mivel a játékmenet mellett most a vizuális elemekre is nagyobb figyelmet kell fordítanom, és együtt kell működnöm egy más emberrel, hogy a játék mind vizuálisan, mind technikailag sikeres legyen. Az együtt működés lehetőséget ad arra hogy jobban megismerjem vizuális elemek kezelését, mivel eddig nem sokat dolgoztam grafikákkal.

Forráskód elérhetősége: github.com/bartusjani/W5OLP9_szakdolgozat

Játék elérhetősége: example.com

Bemutató videó elérhetősége: example.com

1. fejezet

Technológiai áttekintés

1.1. Godot Engine

A Godot Engine egy nyílt forráskódú és ingyenes . Általános célú 2D és 3D játékmotor, ami mindenféle projektet támogat. Lehetővé teszi a játékok kiadását különböző platformokra. A Godotban lehet C#,C++ vagy GDScripttel programozni.

1.1.1. GDScript nyelv

A GDScript Godot specifikus nyelv. Ez a programozási nyelv egy objektumorientált, imperatív, magas szintű nyelv. A Pythonhoz hasonló behúzásalapú szintaxist használ. A Godot Enginehez optimalizált és integrált, célja hogy nagy rugalmasságot biztosítson a szoftverfejlesztéshez. A GDScript a Pythontól teljesen független, és nem arra épül.

A GDScript azonosítói kizárólag betűket (a-z, A-Z), számokat (0-9) és aláhúzás jelet tartalmazhatnak. Fontos hogy az azonosító nem kezdődhet számjeggyel.A nyelv kis és nagybetű érzékeny tehát például valtozo és a VALTOZO mást különböző változónak számítanak. Támogatja a UAX#31 szabványú Unicode karaktereket.

Az egész és lebegőpontos számokat aláhúzással (_) elválasztva is lehet írni. Például 123456789-et lehet 123_456_789-nek írni és a nyelv fel fogja ismerni.

A kommentet a Pythonhoz hasonlóan #-el lehet tenni. Lehet a kommentekből régiót csinálni ami össze csukható.A régiót így lehet csinálni #region ... #endregion.

Beépített adattípusok

Alapértelmezés szerint veremalapú objektumokként tárolódnak, amely érték szerint kerülnek átadásra.

Alapvető beépített típusok:

- null
- bool

- int
- float
- String
- StringName
  Egy nem módosítható karakterlánc, ami biztosítja, hogy egy adott szöveg csak egyszer legyen a memóriában. Bár létrehozása erőforrás-igényesebb, gyorsabb összehasonlítást tesz lehetővé, ezért ideális szótárkulcsokhoz.
- NodePath
  Egy előfeldolgozott útvonal csomópontokhoz, amely könnyen átalakítható String típusúvá.

Vektor adattípusok:

- Vektor2, Vektor2i
  2D vektor típus ami x és y mezőt tartalmaz. A Vector2i-nél csak integer lehet az x és y mezőben.
- Vector3, Vector3i
  3D vektor típus ahol x és y mező mellett van y mező is. Itt is csak integer lehet a Vector3i mezőiben.
- Transform2D
  Egy 3×2-es mátrix, ami 2D transzformációk végrehajtására alkalmas.
- Transform3D
  Egy 3D transzformációt reprezentáló típus, amely egy Basis mezőből és egy Vector3 mezőből áll.
- Basis
  Egy 3x3-as mátrix amit 3D forgatás és skálázásra használnak.

1.1.2. A Godot verzóinak áttekintése

A Godot Engine-t folyamatosan fejlesztik , rendszeresen új funkciókkal, teljesítménybeli javításokkal és hibajavításokkal frissül.Az alábbiakban a legfontosabb verziókról fogok írni.

- Godot 1.0
  2014 decemberében jelent meg a Godot Engine első stabil verziója. Több száz hibát javítottak és a közösség is jelentősen megnövekedett.

- Godot 2.0
  2016 februárjában jelent meg.A Godot 2.0-ban javították jelenetpéldányosítást.Bevezették a jelenet öröklést és egy új szöveges jelenetformátumot ami könnyeben kezelhető, Git kompatibilis és gyorsabb. Továbbá támogatja az onready kulcsszót és singeltonokat.

- Godot 3.0
  2018 januárjában jelent meg. A Godot 3.0-ban új fizikai alapú 3D renderelő kapott helyet. Behozták a GDNative-ot ami egy új keretrendszer amivel könnyen bővíthető a Godot C/C++ nyelven a motor újrafordítás nélkül.

- Godot 4.0
  2023 márciusában jelent meg és jelentős javításokat hozott . A Godot 4.0-ban a 2D munkafolyamatokban új tilemap szerkesztőt vezettek be amivel könnyebb a szint tervezés. A 3D területén a shader-ek és VFX rendszerek újításokon estek át, emellett jelentős fejlesztéseket kapott és shader szerkesztő is.

Most a legfrissebb verziója a Godot 4.4.1 amit 2025 márciusában adtak ki.

1.1.3. Licenszi kérdések

A Godot az MIT licenc alatt készült és kerül kiterjesztésre. Az MIT licenc egyetlen követelménye, hogy a licenc szövegét valahol a játékban el kell helyezni.

1.1.4. A motor fő erősségei

- Intuitív jelenetvezérelt tervezés
  A játékokat egyszerű blokkokból építheted fel, ahol a csomópontok (nodes) hierarchiája segít az átlátható kialakításában.

- Testre szabott kódolási eszközök
  A GDScript és C# nyelvek biztosítanak gyors fejlesztést, míg a Godot 4.0 új statikus típusellenőrzése növeli a hatékonyságot és teljesítményt.

- Egyszerű, mégis nagy teljesítményű 3D motor
  Támogatja a magas és az alacsony teljesítményű eszközöket, a Vulkan renderelő kiaknázza a játék GPU-k erejét.

- Speciális 2D munkafolyamat játékokhoz és alkalmazásokhoz
  A dedikált 2D tile map editor lehetővé teszi a gyors világépítést, egyszerűsíti a logikát és a GUI rendszert a játékokhoz.

1.2. Unity

1.3. Unreal Engine

Az Unreal Engine az Epic Games által fejlesztett, nagy teljesítményű játékmotor.Az első verzióját 1998-ban adták ki. A legfrissebb verziója az Unreal Engine 5. A motor támogatja a C++ programozási nyelven való fejlesztést, de a Blueprint rendszere lehetővé teszi a vizuális programozást is. A motor teljesen ingyen használható egy bevételi határ alatt, így lehetőséget ad a kisebb fejlesztők számára is.

1.3.1. Blueprint

A Blueprint a játékmotor egyik fontos eszköze, amely lehetővé teszi a játékok,alkalmazások logikájának vizuális szkriptelését. Különösen hasznos a nem programozó felhasználóknak, mivel így mélyebb programozási tudás nélkül is képesek a projektjeiket megvalósítani.

A vizuális programozás erősségei

A Blueprint egy teljes játékmenet–szkriptrendszer, amely csomópont–alapú koncepción alapul. Objektumorientált osztályok vagy objektumok definiálására szolgál a motorban.Ez a rendszer rendkívül rugalmas és nagy teljesítményű,lehetővé teszi a tervezők számára, hogy az általában csak a programozók számára elérhető fogalmak és eszközök teljes skáláját használhassák.

Számos előnyt kínál a fejlesztés során. Lehetővé teszi az egyszerűbb definiálását a viselkedéseknek, ami megkönnyíti a szkriptelést és a projekt logikájának kialakítását. Ideális eszköz a prototípusok készítésére, mivel gyorsítja az osztályok létrehozását,módosítását,lefordítását és tesztelését ezzel jelentős időt spórol meg.Az API-k gyorsabb felfedezését is lehetővé teszi.

Blueprint osztályok

A Blueprint osztály egy olyan eszköz, ami lehetővé teszi a új funkciók hozzáadását a játékmenethez kód írása nélkül.Eszközökként mentődnek el a tartalom csomagban.

- Data–Only Blueprint
  Ez egy olyan osztály ami csak örökölt kódot, változókat, komponenseket tartalmaz. Lehetővé teszi ezek változtatását, de új elemeket nem lehet hozzáadni. Archetípusok helyettesítésére szolgál. Teljes Blueprintté alakítható ha kódot, változókat vagy komponenseket adunk hozzá.

- Level Blueprint
  A Level Blueprint egy speciális Blueprint típus, amely a teljes szint globális eseménygrafikájaként működik. Ez a Blueprint eseményeket, műveleteket kezel szintben lévő szereplőkkel kapcsolatban.

- Blueprint Utilities
  A Blueprint Utility csak a szerkesztőre korlátozódik.Lehetővé teszi a szerkesztőben különböző műveletek végrehajtását vagy funkciók hozzáadását a szerkesztőben. Gyakran használják szkriptek, szerkesztőbeli kiegészítők létrehozására.

1.3.2. Unreal Engine verzióinak áttekintése

A Unreal Engine-t folyamatosan fejlesztik , rendszeresen új funkciókkal, teljesítménybeli javításokkal és hibajavításokkal frissül.Az alábbiakban pár újabb verzióról fogok írni.

- Unreal Engine 4.27
  2021 augusztusában jelent meg. Ez a verzió mindenki számára kínál valamit kezdve a filmesektől, vizualizációs szakembereken át a játékfejlesztőkig. Ebben a kiadásban a munkafolyamatok egyszerűsítésén és a teljesítmény növelésén volt a hangsúly.

- Unreal Engine 5.0
  2022 áprilisában jelent meg. Ezen kiadásban a lehetővé tették a fejlesztők számára hogy next-gen valós idejű 3D tartalmakat,élményeket hozzanak létre nagyobb szabadsággal,rugalmassággal és részletességgel.Az új funkciók mint a Nanite vagy a Lumen új vizuális minőséget biztosítanak, lehetővé téve a dinamikus világok létrehozását.

- Unreal Engine 5.3
  2023 szeptemberében jelent meg. Ez a kiadás tovább javította a UE5 eszközöket.Fejlesztették a renderelést, világépítést, procedurális tartalomgenerálást,animációs és modellezési eszközöket, a szimulációkat. Ezáltal az Unreal Engine 5.3 kiadás tovább erősítette az Epic Games elkötelezettségét a valós idejű grafika és fejlesztői eszközök élvonalbeli fejlesztése iránt.

- Unreal Engine 5.5
  2024 novemberében jelent meg. Ez legfrissebb verziója az Unreal Engine-nek. Jelentős előrelépések történtek az animációk készítésében, a virtuális produkcióban és a mobiljáték–fejlesztésben területén is.Több funkció (például az in–camera–VFX, fejlesztői iteráció) elérte a gyártáskészséget.

1.3.3. Licenszi kérdések

Az Unreal Engine-t az Egpic Games licencfeltételei alapján használható.Alapvetően ingyenesen elérhető, viszont egy meghatározott bevételi küszöb felett a fejlesztő köteles fizetni a játékmotor használatáért.Az alábbi felsorolásban ismertetem a különböző licenc lehetőségeket.

- 1 millió dollár alatti bevétel esetén
  Ingyenes a játék fejlesztők, egyéni fejlesztők, kisebb cégek,valamint iskolák és oktatók számára.

- 1 millió dollár feletti bevétel esetén
  Kétféle fizetési lehetőség közül választhat a fejlesztő:

  - jogdíj alapú fizetés
    Ha olyan játékot vagy alkalmazást készít a fejlesztő, amely futásidőben használja az Unreal Engine kódját, és harmadik fél számára kerül licencelésre, akkor kell jogdíjat fizetnie.
    Költség: Az 1 millió dollárt meghaladó bevétel után 5% jogdíjat kell fizetni.
  - Ülőhely alapú fizetés
    Ha az Unreal Engine-t kereskedelmi célokra használja a fejlesztő,és az elmúlt 12 hónapban több mint 1 millió dollár bevételt termelt,és nem olyan játékot vagy alkalmazást készít, amely futásidőben használja az engine kódját és harmadik félnek licencelhető, akkor ülőhely(seat) licenc díjat kell fizetnie.
    Költség: Évi 777 549 Ft / ülőhely.

1.3.4. A motor fő erősségei

Az Unreal Engine számos erősséggel rendelkezik, amelyek hozzájárulnak ahhoz, hogy az egyik legnépszerűbb játékmotor legyen a piacon. Íme néhány főbb erőssége:

- Grafikai teljesítmény
  Kiváló vizuális minőséget kínál, beleértve a valós idejű ray tracinget, amely valósághű fény- és árnyékhatásokat biztosít.

- Blueprint rendszer
  A vizuális szkriptnyelv lehetővé teszi, hogy kódolás nélkül fejlesszünk játékokat, így gyors prototípusokat készíthetünk.

- Platform támogatás
  Az Unreal Engine támogatja a legtöbb nagy platformot, beleértve a PC-t, konzolokat, mobil eszközöket és VR/AR eszközöket. Ez lehetővé teszi a fejlesztők számára, hogy széles körű közönséghez juttassák el alkotásaikat.

2. fejezet

Rendszerterv

3. fejezet

Saját projekt fejlesztése

4. fejezet

Tesztelés

Összegzés

Ábrák jegyzéke

Irodalomjegyzék

Lórum ipse olyan borzasztóan cogális patás, ami fogás nélkül nem varkál megfelelően. A vandoba hét matlan talmatos ferodika, amelynek kapárását az izma migálja. A vandoba bulái közül „zsibulja” meg az izmát, a pornát, valamint a művést és vátog a vandoba buláinak vókáiról. Vókája a raktil prozása két emen között. Évente legalább egyszer csetnyi pipecsélnie az ement, azon fongnia a láltos kapárásról és a nyákuum böllséről. A vandoba ninti és az emen elé redőzi a szamlan radalmakan érvést. Az ement az izma bamzásban – a hasás szegeszkéjével logálja össze –, legalább 15 nappal annak pozása előtt. Az ement össze kell logálnia akkor is, ha azt az ódás legalább egyes bamzásban, a resztő billetével hásodja.

Irodalomjegyzék

[1] Fazekas István: *Valószínűségszámítás*, Debreceni Egyetem, Debrecen, 2004.

[2] Tómács Tibor: *A valószínűségszámítás alapjai*, Líceum Kiadó, Eger, 2005.

Nyilatkozat

Alulírott ....., büntetőjogi felelősségem tudatában kijelentem, hogy az általam benyújtott, ... című szakdolgozat önálló szellemi termékem. Amennyiben mások munkáját felhasználtam, azokra megfelelően hivatkozom, beleértve a nyomtatott és az internetes forrásokat is.

Aláírásommal igazolom, hogy az elektronikusan feltöltött és a papíralapú szakdolgozatom formai és tartalmi szempontból mindenben megegyezik.

Eger, 2024. december 18.

aláírás

A szakdolgozat megírása után ezt a nyilatkozatot kell a végéhez csatolni a következő módon:

1. A `nyilatkozat` mappában a `nyilatkozat.tex` fájlt töltse ki és fordítsa le pdf-be!
2. A `nyilatkozat.pdf` fájlt nyomtassa ki, majd írja alá!
3. Szkennelje be pdf formátumba, majd ezzel írja felül a `nyilatkozat` mappában a `nyilatkozat.pdf` fájlt!
4. A szakdolgozat forrásfájljában legyen betöltve a `pdfpages` csomag. Az utolsó két sor legyen az alábbi:

   ```
   \includepdf{nyilatkozat/nyilatkozat.pdf}
   \end{document}
   ```

5. Ezután fordítsa le a szakdolgozatot pdf-be!